¿Es verdad que la programación orientada a objetos solo complica las cosas innecesariamente y hace que el software sea más lento?

Teóricamente añade una capa de abstracción lo que implica más complejidad y más costos operativos. Pero la ganancia es lo cotrario de complicar: simplificar la solución al brindar herramientas cognitivas a los programadores.

Pero … sí que puede llegar a ser un estorbo para los programadores, causando diseños complicados, si se insiste en utilizar ese paradigma en dominios donde no es adecuado.

Los lenguajes son sólo herramientas, y no debes ver todo problema como un clavo.

Nada mejor que la POO para el desarrollo de aplicaciones interactivas con metáforas de la vida real (escritorio, ventana, icono, etc.) ~y/o en tiempo real (Entornos de Escritorio, video juegos, etc.)~.

En otros dominios, la POO empieza a perder su pertinencia. Un buen ejemplo es el procesamiento y análisis de datos, en el que la programación funcional es más adecuada e insistir en un diseño bajo el paradigma POO entorpecería el proceso de desarrollo.

———

Actualización:

Inicialmente mencioné que la POO era el mejor estilo de programación para, entre otras cosas, programar video juegos.

Sin embargo, he aprendido que, a pesar de ser el paradigma preferido en la industria de los videojuegos, la POO es un problema gratis que los diseñadores tienen que resolver luego con técnicas de agrupación de datos, sobre todo en el caso de videojuegos en tiempo real que requieren un alto desempeño de la computadora.

Si el diseño del juego se centra en los datos y sus transformaciones, estos problemas de desempeño que introduce la POO se evitarían. Y por supuesto, en este tipo de diseños centrados en los datos, es más intuitivo utilizar la programación funcional.

Comparto un artículo (en inglés) que explica el concepto del Diseño Orientado a los Datos[1].

Notas al pie

[1] Data-Oriented Design (Or Why You Might Be Shooting Yourself in The Foot With OOP)